Introduction

Readies Payment for Wix is a service plugin that lets you accept payments on your Wix website.
Requirements

- Your website must have â€œWix Storesâ€ enabled.
- Your website must include the â€œWix Memberâ€ Area.

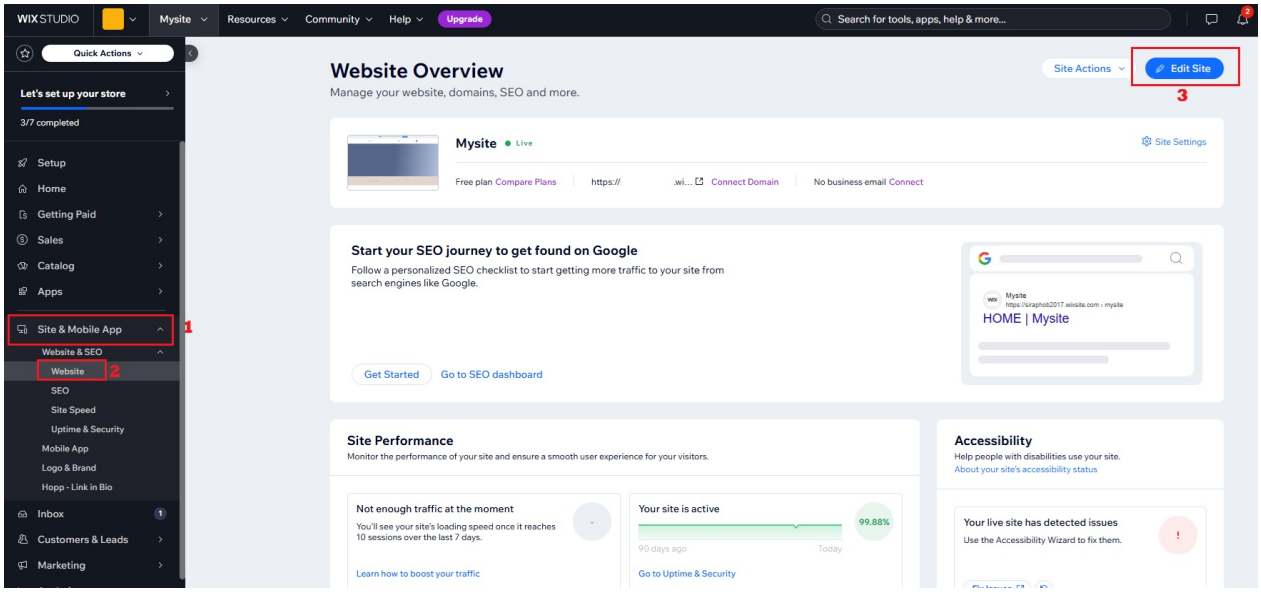
**For detailed steps on how to integrate a service plugin with Wix, refer to the official Wix documentation **. Click here

---

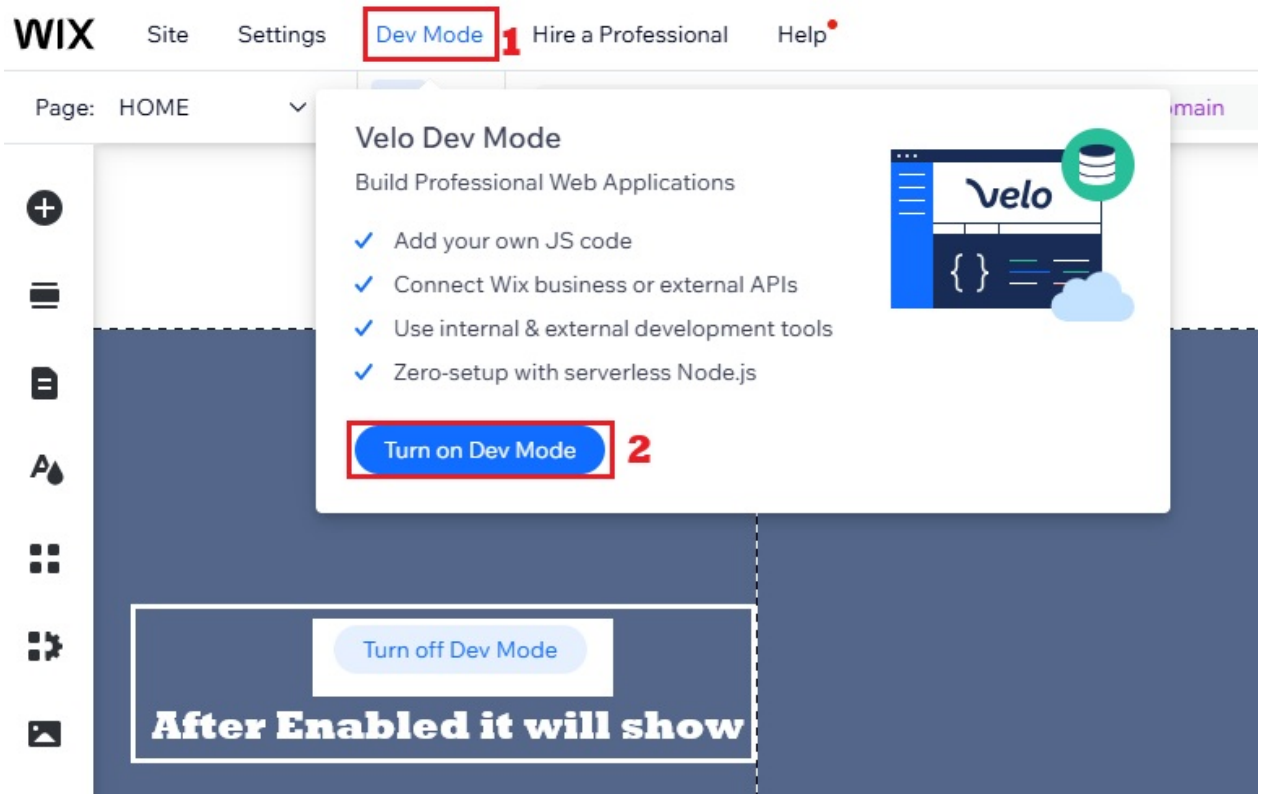
Add New Payment

Step 1 : Add a New Payment Service Plugin to Your Site

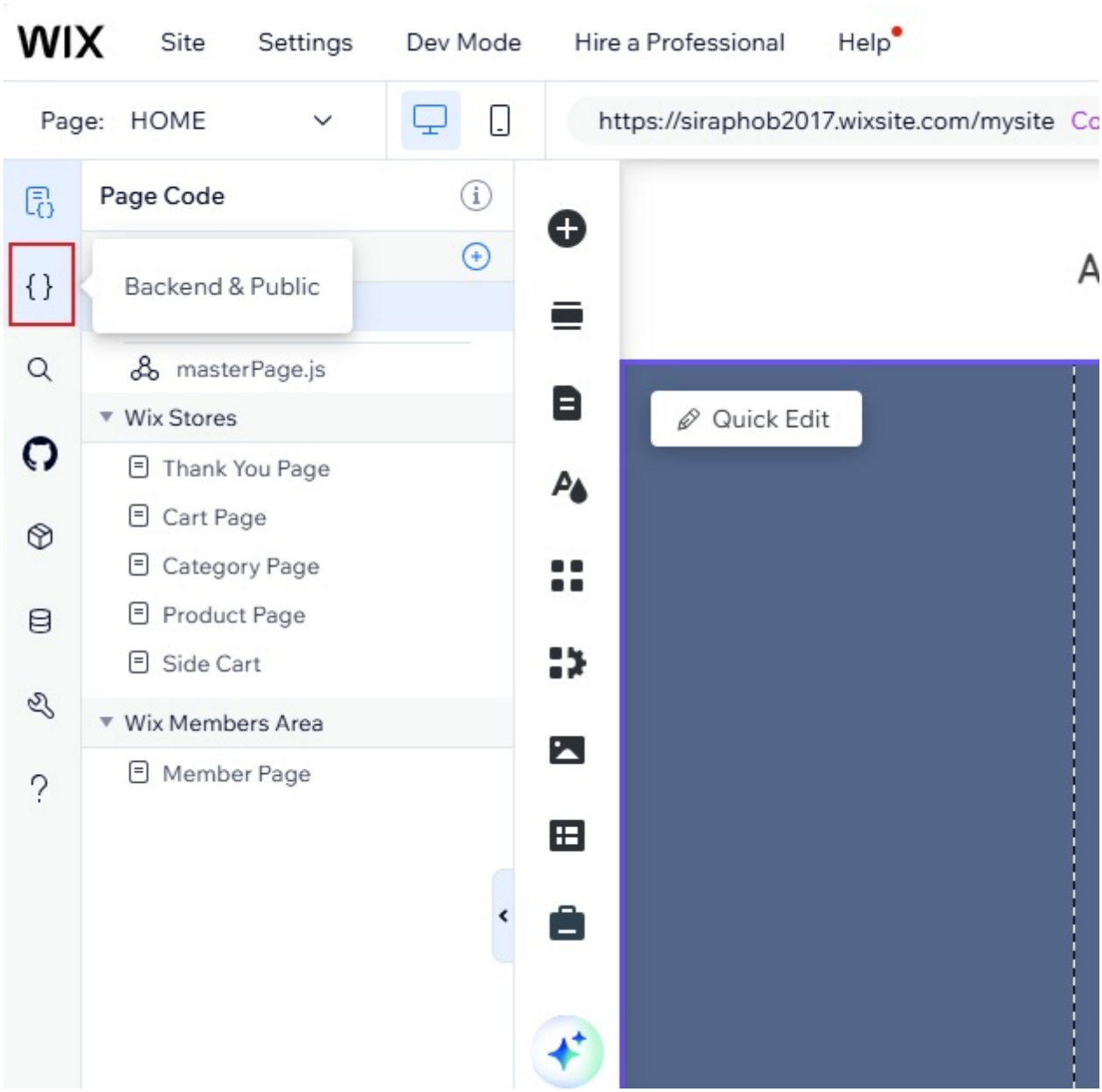
- Open your site editor.



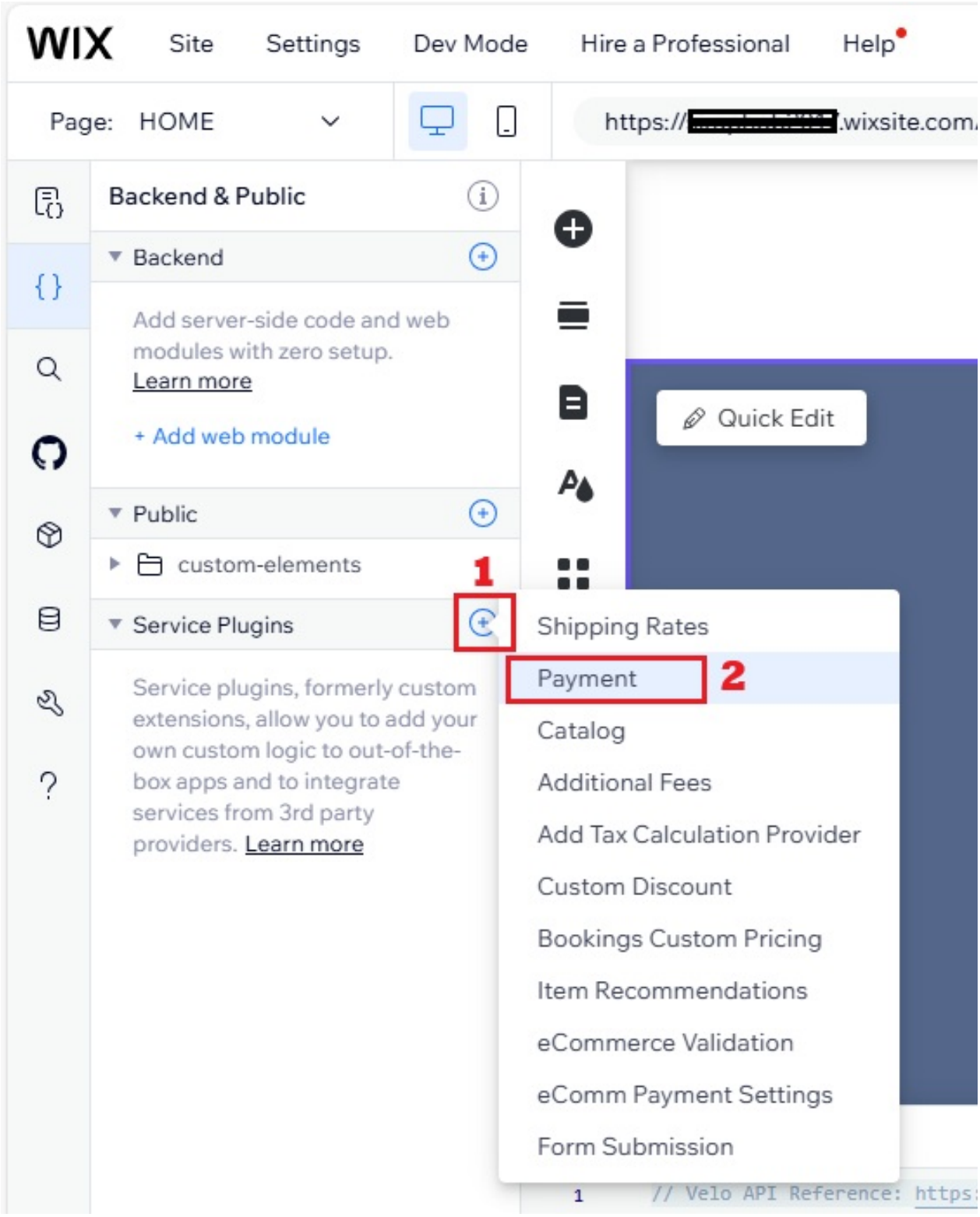
- Enabled Dev Mode



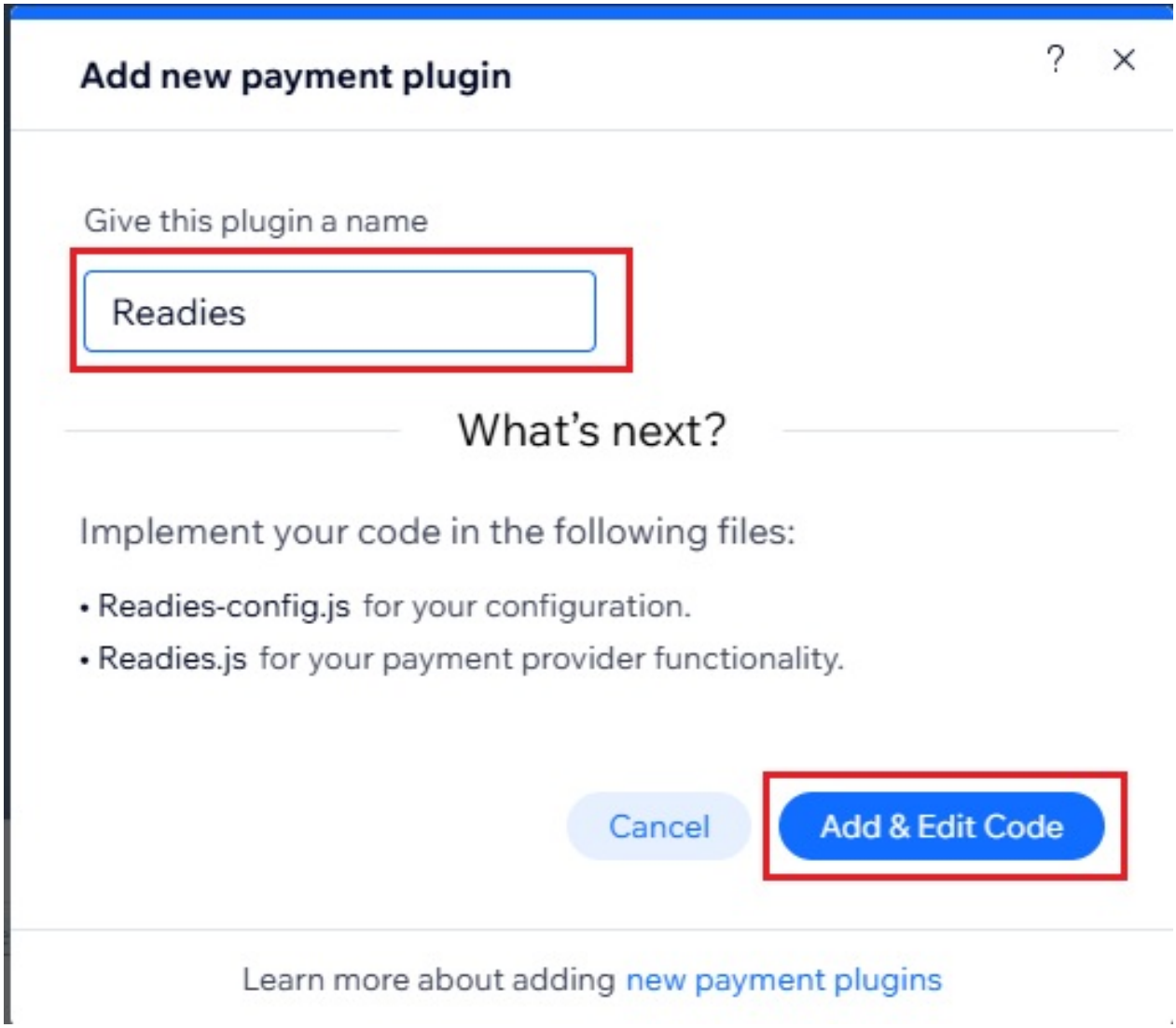
- Go to Backend & Public > Service Plugins.



- Click "Add Payment Service Plugin".



- Enter the name of your new payment plugin.
- Click "Add & Edit Code" to create the plugin.



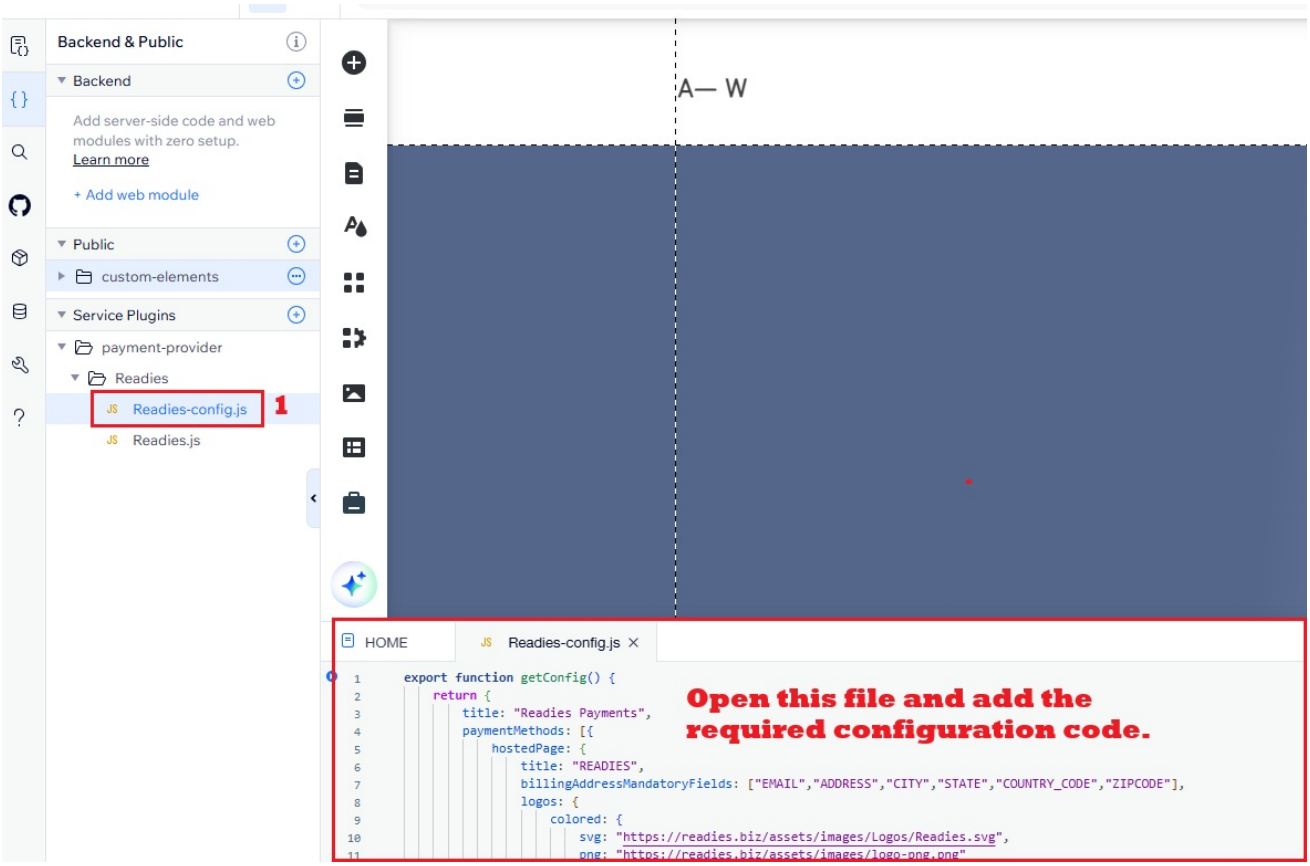
Step 2 : Configure the Plugin Files

Once the plugin is created, Wix will generate two files for you:

- {plugin name}-config.js
- {plugin name}.js

1. Edit {plugin name}-config.js

Open this file and add the required configuration code.

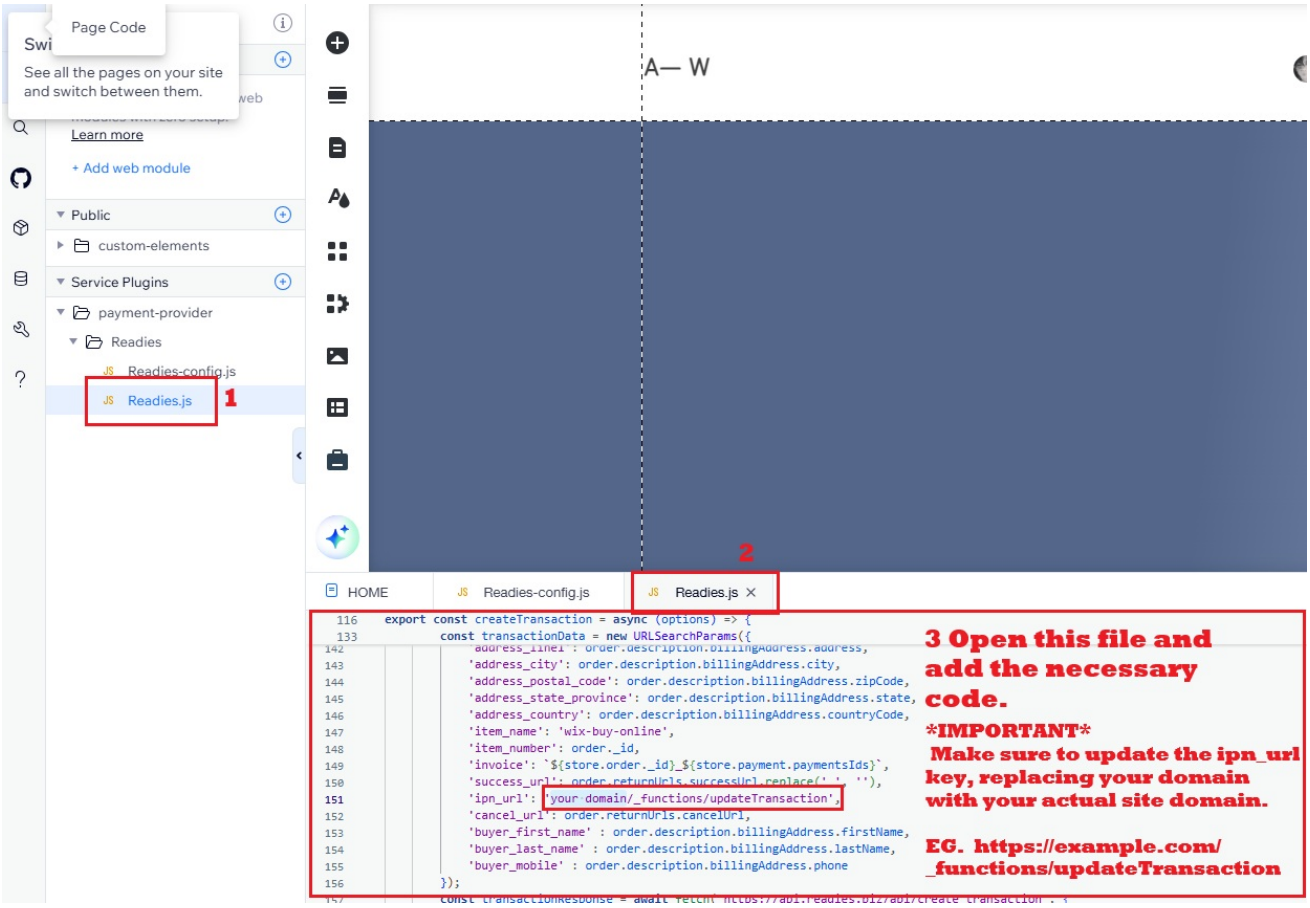



```
export function getConfig() {
    return {
        title: "Readies Payments",
        paymentMethods: [{
            hostedPage: {
                title: "READIES",
                billingAddressMandatoryFields: ["EMAIL","ADDRESS","CITY","STATE","COUNTRY_CODE","ZIPCODE"],
                logos: {
                    colored: {
                        svg: "https://readies.biz/assets/images/Logos/Readies.svg",
                        png: "https://readies.biz/assets/images/logo-png.png"
                    }
                }
            }
        }],
        credentialsFields: [
            {
                simpleField: {
                    name: "email",
                    label: "Merchant Email"
                }
            },
            {
                simpleField: {
                    name: "public_key",
                    label: "Public Key"
                }
            },
            {
                simpleField: {
                    name: "private_key",
                    label: "Private Key"
                }
            }
        ]
    };
}
```

2. Edit {plugin name}.js

Open this file and add the necessary code.

IMPORTANT Make sure to update the ipn_url key, replacing **your domain** with your actual site domain.



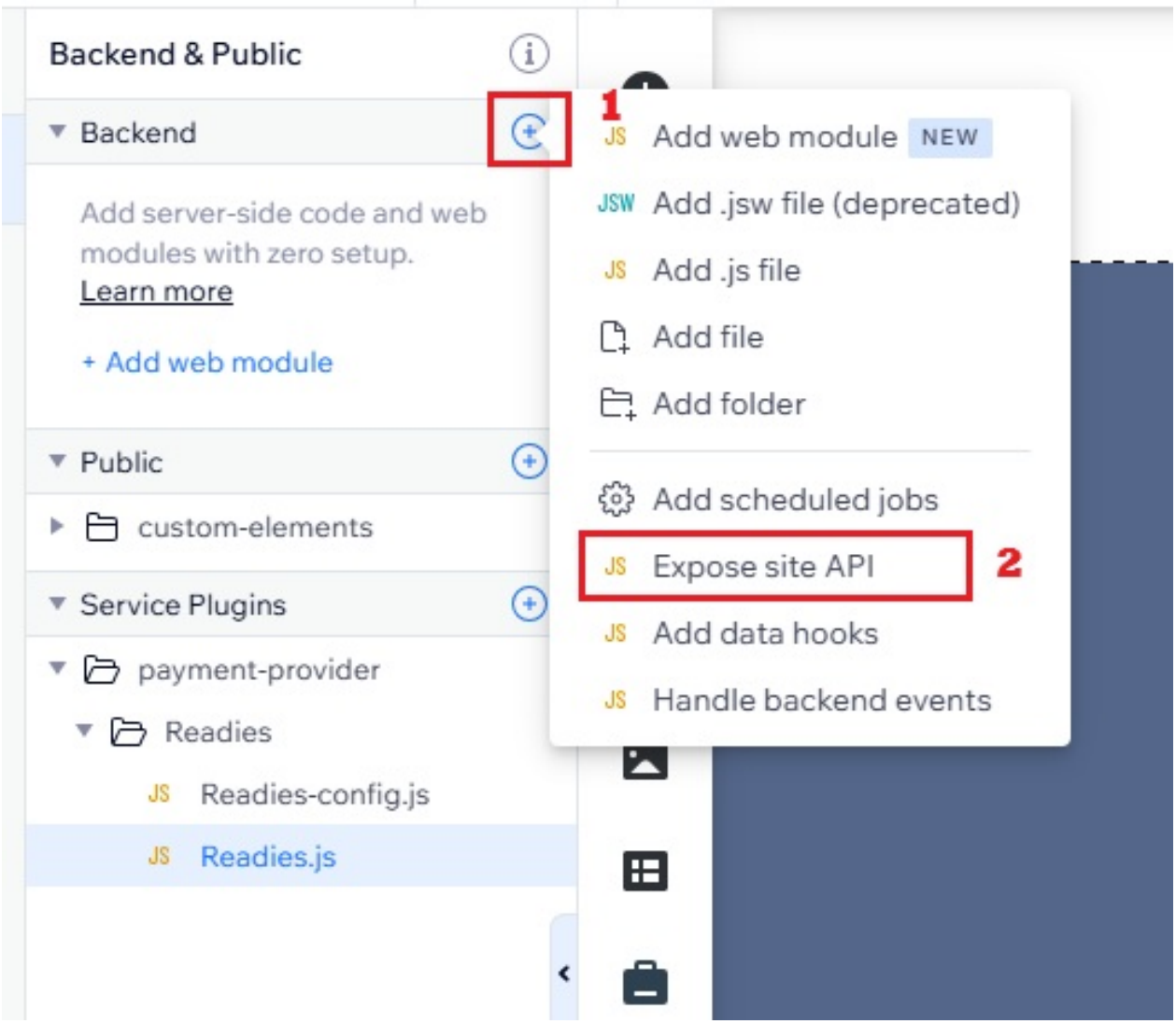

```
import { fetch } from 'wix-fetch';
import wixUsersBackend from 'wix-users-backend';
import wixStoresBackend from "wix-stores-backend";
import { orderTransactions } from "wix-ecom-backend";
import { elevate } from "wix-auth";

const handleApiResponse = async (response) => {
    if (!response.ok) {
        const errorText = await response.text();
        throw new Error(`API request failed: ${response.status} - ${errorText}`);
    }
    return await response.json();
};

export const connectAccount = async (options) => {
    const { credentials } = options;
    const connectData = new URLSearchParams(credentials);
    try {
        const apiResponse = await fetch("https://api.readies.biz/api/connectAccount", {
            method: "POST",
            headers: { 'Content-Type': 'application/x-www-form-urlencoded' },
            body: connectData.toString(),
        });
        const authJson = await handleApiResponse(apiResponse);
        if (authJson.credentials?.client_id) {
            return {
                credentials: {
                    public_key: authJson.credentials.client_id,
                    private_key: authJson.credentials.client_secret,
                    accountName: authJson.accountName,
                },
            };
        }
        return { error: "Merchant not found" };
    } catch (error) {
        console.error('Connect error:', error);
        return { error: error.message };
    }
};

export const storeOrder = async (options) => {
    const { order } = options;
    const currentUser = wixUsersBackend.currentUser;
    const currentUserId = currentUser.id;
    const billingAddress = order.description.billingAddress;
    const items = order.description.items.map(item => ({
        lineItemType: "PHYSICAL",
        name: item.name,
        priceData: { price: item.price },
        productId: item._id,
        quantity: item.quantity,
        productName: item.name,
    }));
    const orderData = {
        cartId: order._id,
    billingInfo: {
        address: {
```

```
                    addressLine: billingAddress.address,
                    city: billingAddress.city,
                    postalCode: billingAddress.zipCode,
                    subdivision: billingAddress.state,
                    country: billingAddress.countryCode,
                },
                firstName: billingAddress.firstName,
                lastName: billingAddress.lastName,
                email: billingAddress.email,
            },
            shippingInfo: {
                shipmentDetails: {
                    address: {
                        addressLine: billingAddress.address,
                        city: billingAddress.city,
                        postalCode: billingAddress.zipCode,
                        subdivision: billingAddress.state,
                        country: billingAddress.countryCode,
                    },
                },
                firstName: billingAddress.firstName,
                lastName: billingAddress.lastName,
                email: billingAddress.email,
            },
            buyerInfo: {
                email: billingAddress.email,
                firstName: billingAddress.firstName,
                lastName: billingAddress.lastName,
                phone: billingAddress.phone,
                id: currentUserId,
                identityType: "MEMBER",
            },
            buyerLanguage: 'en',
            channelInfo: { type: 'WEB' },
            currency: order.description.currency,
            lineItems: items,
            paymentStatus: "NOT_PAID",
            totals: {
                total: parseFloat(order.description.totalAmount) / 100,
                subtotal: parseFloat(order.description.totalAmount) / 100,
            },
        };

        try {
            const createdOrder = await wixStoresBackend.createOrder(orderData);
            const payment = [{
                amount: { amount: createdOrder.totals.total.toString() },
                refundDisabled: true,
                regularPaymentDetails: { offlinePayment: false },
            }];
            const elevatedAddPayments = elevate(orderTransactions.addPayments);
            const paymentResult = await elevatedAddPayments(createdOrder._id, payment);
            return { payment: paymentResult, order: createdOrder };
            } catch (error) {
                console.error("Error while creating payment or order:", error);
                throw error;
            }
        };

        export const createTransaction = async (options) => {
            const { merchantCredentials, order } = options;
            const { accountName, public_key, private_key } = merchantCredentials;
            const credential = new URLSearchParams({ email: accountName, public_key, private_key });
            try {
                const authResponse = await fetch("https://api.readies.biz/api/get_authorized_token", {
                    method: "POST",
                    headers: { 'Content-Type': 'application/x-www-form-urlencoded' },
                    body: credential.toString(),
                });
                const authJson = await handleApiResponse(authResponse);
                const token = authJson.response?.authorize_token;
                if (!token) {
                    return { reasonCode: 6000, errorCode: "General error", errorMessage: "Merchant API token Error" };
                }
                const conAmount = (parseFloat(order.description.totalAmount) / 100).toFixed(2);
                const store = await storeOrder(options);
                const transactionData = new URLSearchParams({
                    'cmd': 'simple',
                    'amount': conAmount,
                    'currency1': order.description.currency,
                    'currency2': 'READIES',
                    'user_creation': 'true',
                    'buyer_opt_completed': '0',
                    'buyer_email': order.description.billingAddress.email,
                    'address_supplied': 'true',
                    'address_line1': order.description.billingAddress.address,
                    'address_city': order.description.billingAddress.city,
                    'address_postal_code': order.description.billingAddress.zipCode,
                    'address_state_province': order.description.billingAddress.state,
                    'address_country': order.description.billingAddress.countryCode,
                    'item_name': 'wix-buy-online',
                    'item_number': order._id,
                    'invoice': `${store.order._id}_${store.payment.paymentsIds}`,
                    'success_url': order.returnUrls.successUrl.replace(' ', ''),
                    'ipn_url': 'your domain/_functions/updateTransaction',
                    'cancel_url': order.returnUrls.cancelUrl,
                    'buyer_first_name' : order.description.billingAddress.firstName,
                    'buyer_last_name' : order.description.billingAddress.lastName,
                    'buyer_mobile' : order.description.billingAddress.phone
                });
                const transactionResponse = await fetch("https://api.readies.biz/api/create_transaction", {
```

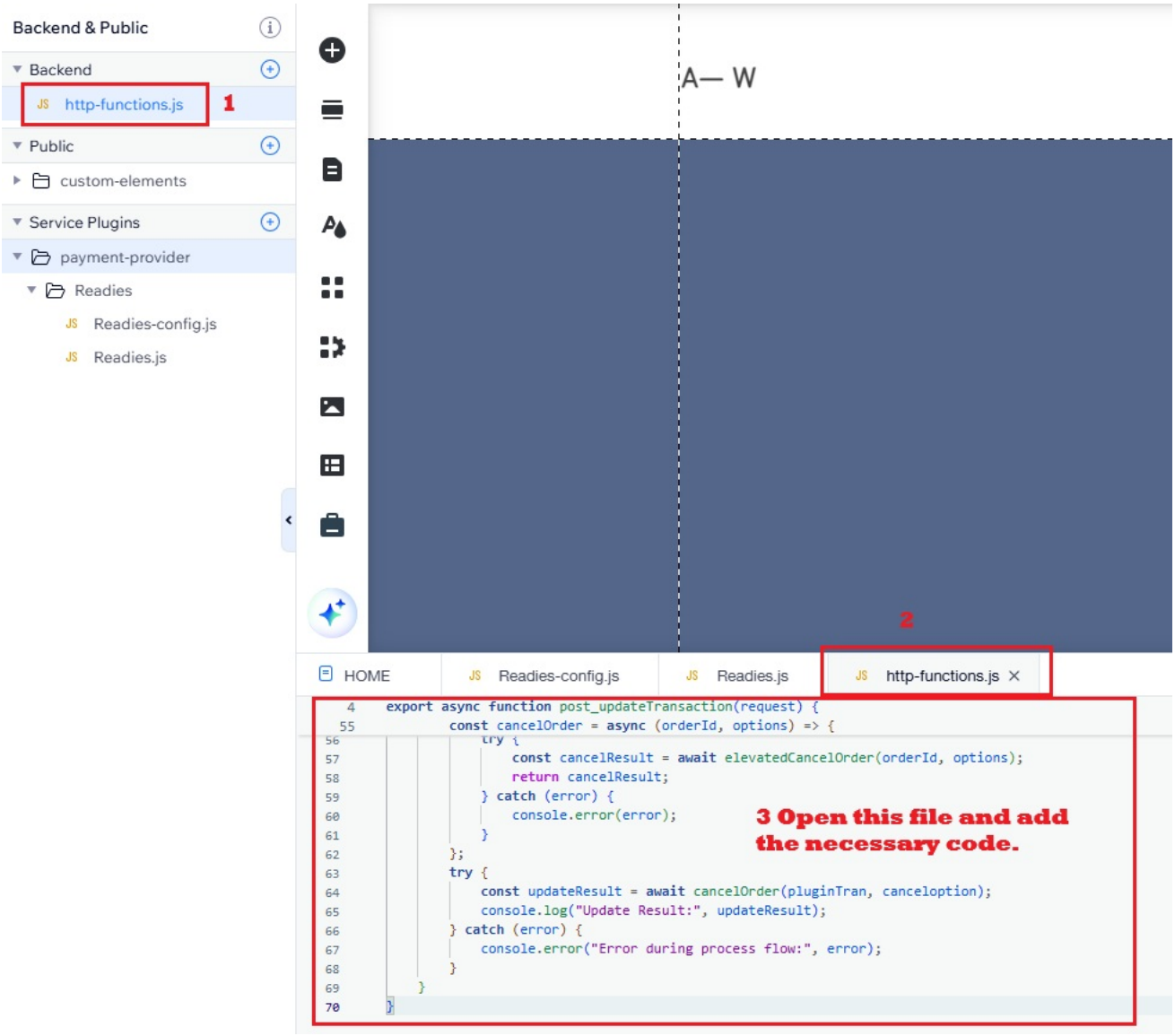
```
            method: "POST",
            headers: { 'Content-Type': 'application/x-www-form-urlencoded', 'Authorization': token },
            body: transactionData.toString(),
        });
        const transactionJson = await handleApiResponse(transactionResponse);
        if (transactionJson.response.checkout_url) {
            return {
                pluginTransactionId: transactionJson.response.payment_id,
                redirectUrl: transactionJson.response.checkout_url,
            };
        }
        return { reasonCode: 6000, errorCode: "General error", errorMessage: "Create Transaction Error" };
        } catch (error) {
            return { reasonCode: 6000, errorCode: "General error", errorMessage: error.message };
        }
    };
    export const refundTransaction = async () => {
        return {
            reasonCode: 6000,
            errorCode: "General error",
            errorMessage: "Refund not available",
        };
    };
```

Step 3 : Create a New API for Payment Status

- In the Backend folder, click the Create icon and add a new Expose Site API.

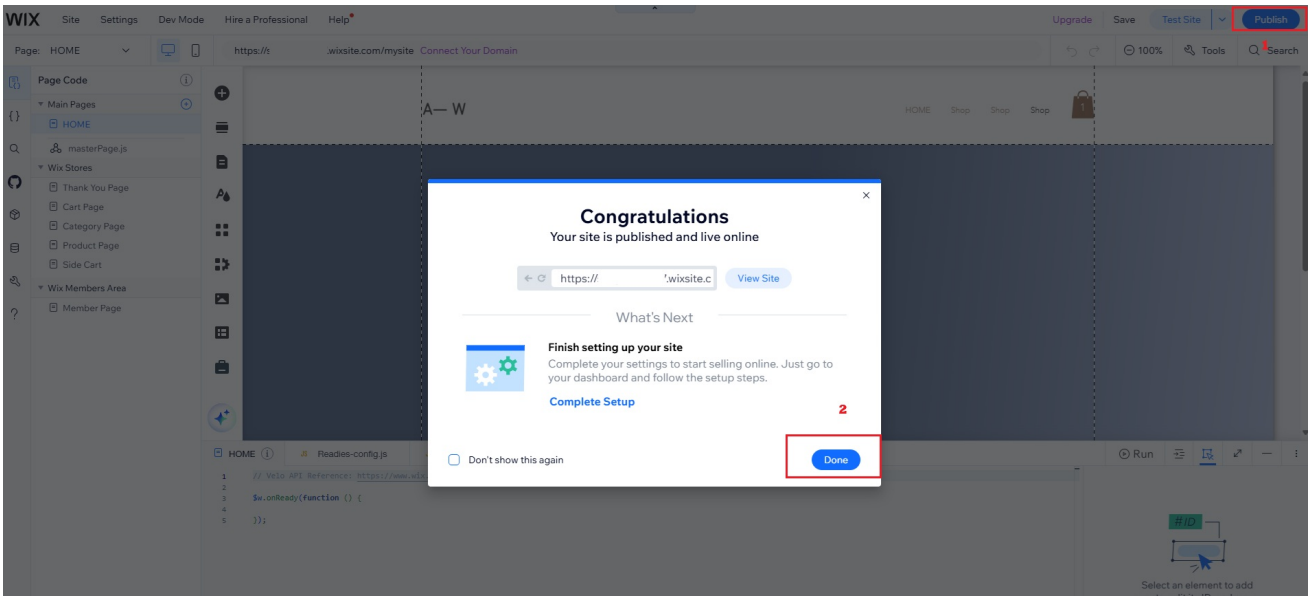


- Add the required code to this API file.
- This API will be used to retrieve payment status from Readies.



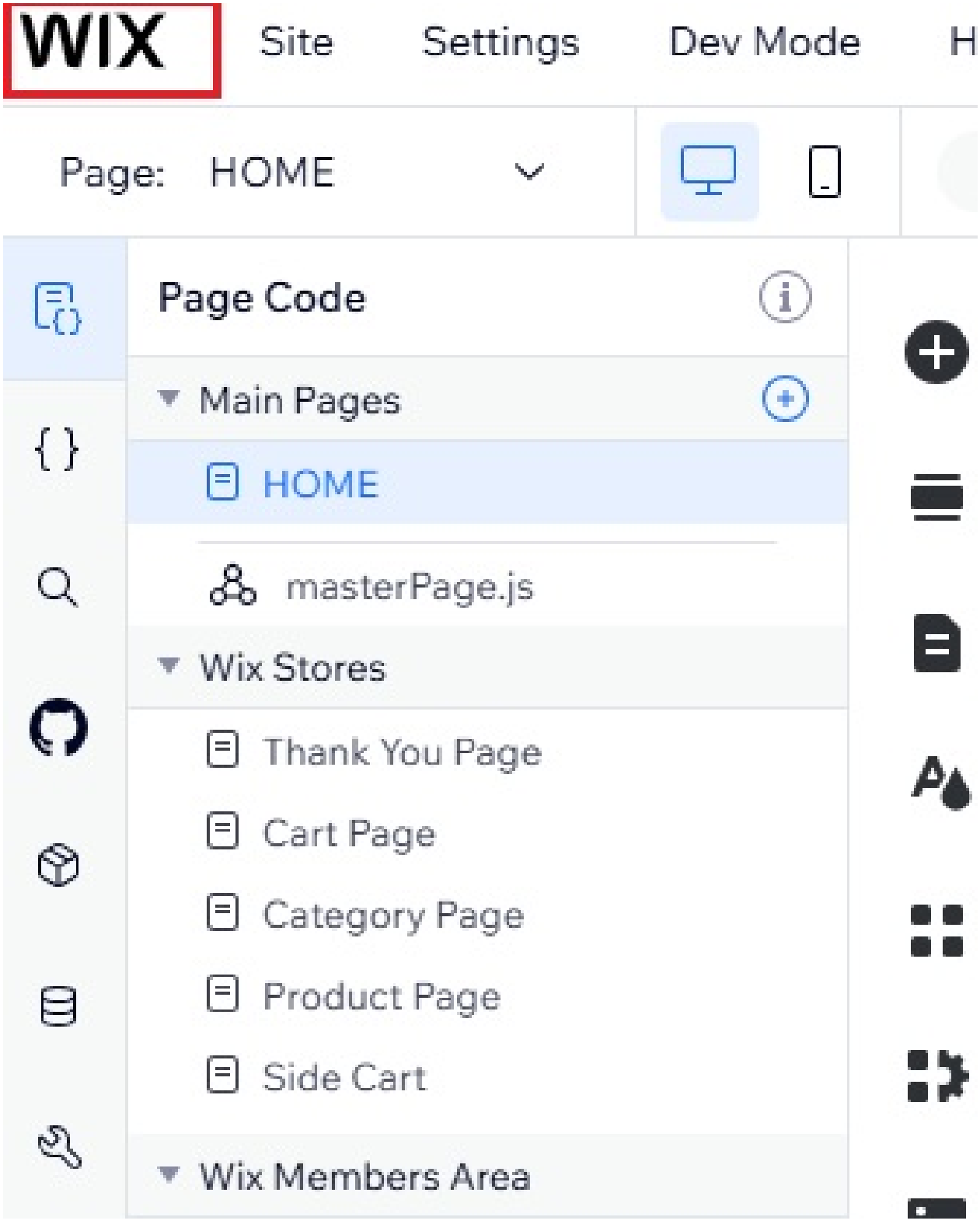

```
import { orderTransactions, orders } from "wix-ecom-backend";
import { elevate } from 'wix-auth';

export async function post_updateTransaction(request) {
    const bodyText = await request.body.text();
    const requestBody = JSON.parse(bodyText);
    let status = requestBody.resultCode;
    let readiesPaymentID = requestBody.payment_id;
    let invoice = requestBody.invoice_id.split('_');
    let pluginTran = invoice[0];
    let paymentId = invoice[1];
    let op = null;

    if (status === 'completed') {
        op = { status: "APPROVED" };
    } else if (status === 'failed') {
        op = { status: "DECLINED" };
    } else if (status === 'cancelled') {
        op = { status: "CANCELED" };
    } else {
        console.error("Unknown status:", status);
    }
    if (op.status == "APPROVED") {
        try {
            const elevatedUpdatePaymentStatus = elevate(orderTransactions.updatePaymentStatus);
            const updatePaymentStatus = async (iden, op) => {
                try {
                    const result = await elevatedUpdatePaymentStatus(iden, op);
                    return result;
                } catch (error) {
                    console.error("Error in updating payment status:", error);
                    throw error;
                }
            };

            if (op) {
                const iden = {
                    orderId: pluginTran,
                    paymentId: paymentId,
                };
                const updateResult = await updatePaymentStatus(iden, op);
                console.log("Update Result:", updateResult);
            } else {
                console.log("No operation to perform (op is null)");
            }
        } catch (error) {
```

```
                    console.error("Error while processing the transaction update:", error);
                }
            } else {
                let canceloption = {
                    customMessage: op.status == "CANCELED" ? "Order Cancelled from Readies" : "Payment Declinded from
                    restockAllItems: true,
                    sendOrderCanceledEmail: true
                }
                const elevatedCancelOrder = elevate(orders.cancelOrder);
                const cancelOrder = async (orderId, options) => {
                try {
                    const cancelResult = await elevatedCancelOrder(orderId, options);
                        return cancelResult;
                    } catch (error) {
                        console.error(error);
                    }
                };
                try {
                    const updateResult = await cancelOrder(pluginTran, canceloption);
                    console.log("Update Result:", updateResult);
                } catch (error) {
                    console.error("Error during process flow:", error);
                }
            }
        }
```

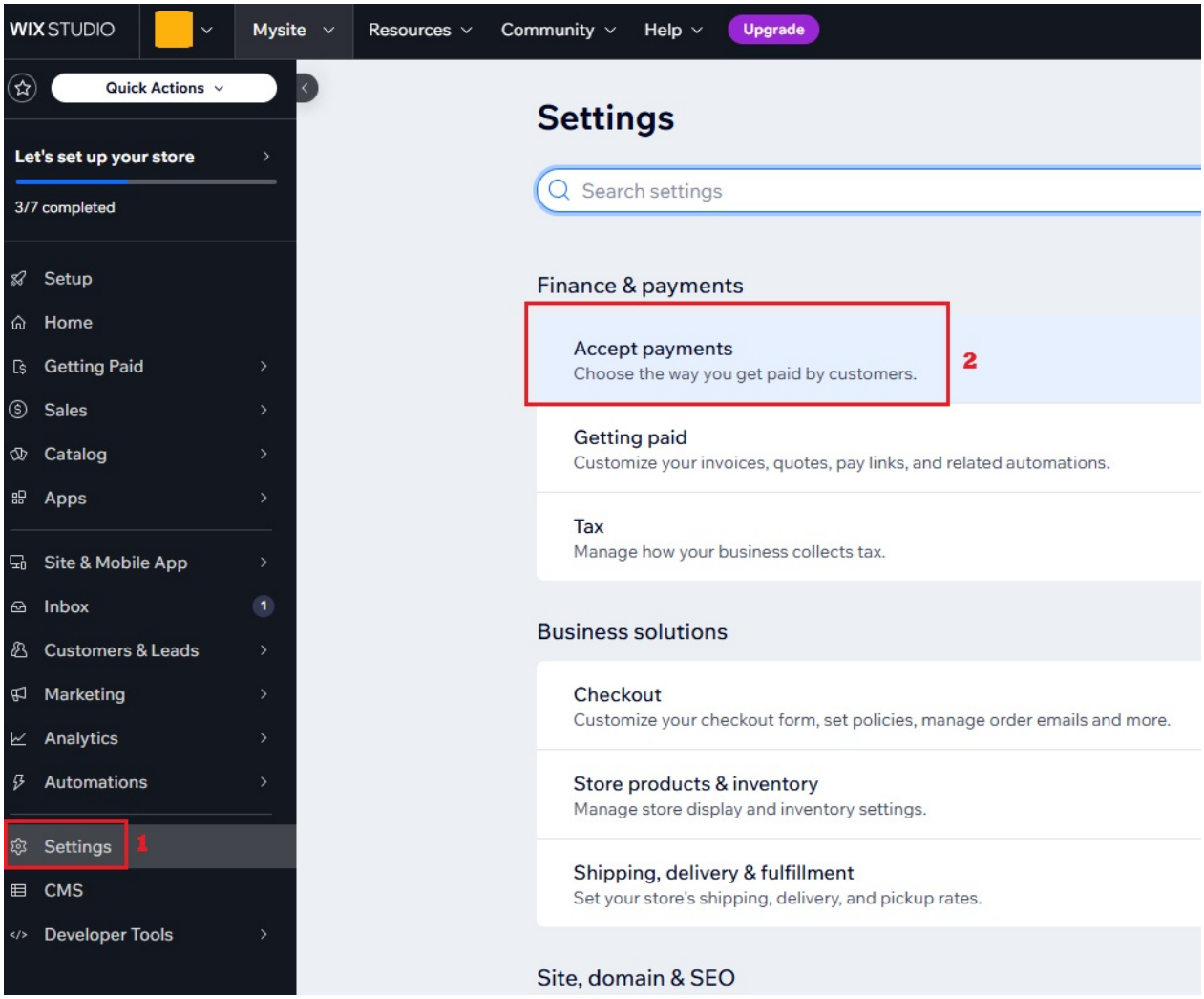


Update Payment
Updating Payment Configuration on your Dashboard. After creating and publishing your new payment service plugin, follow these steps to connect it to your Wix site:

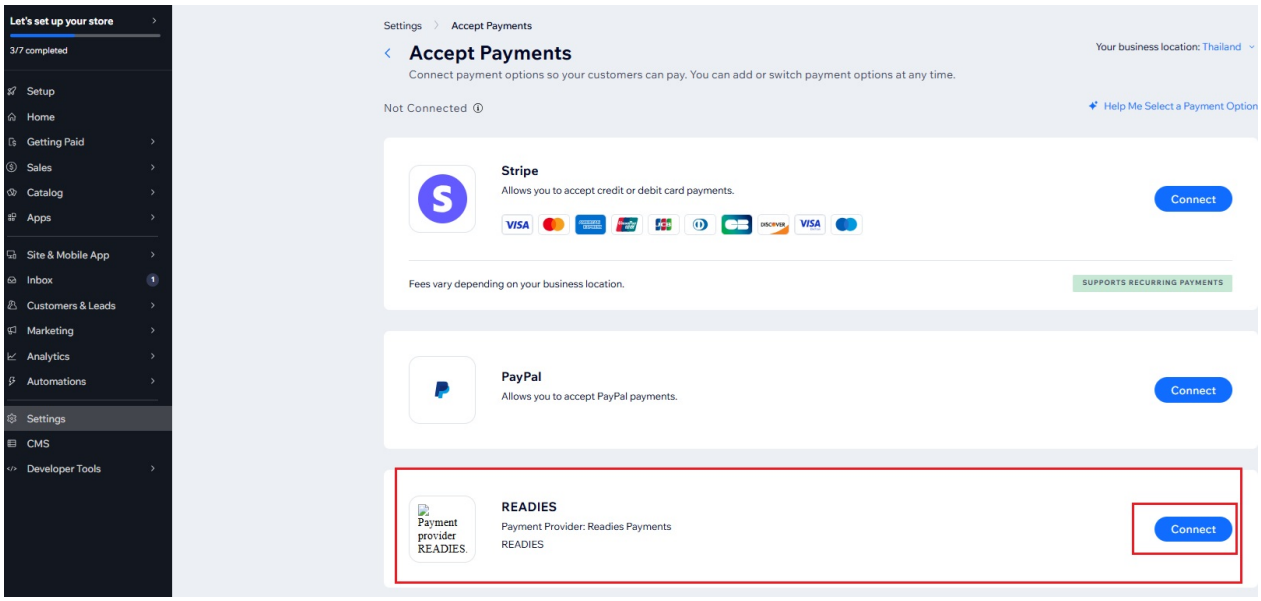
- Go to your Wix website's settings.



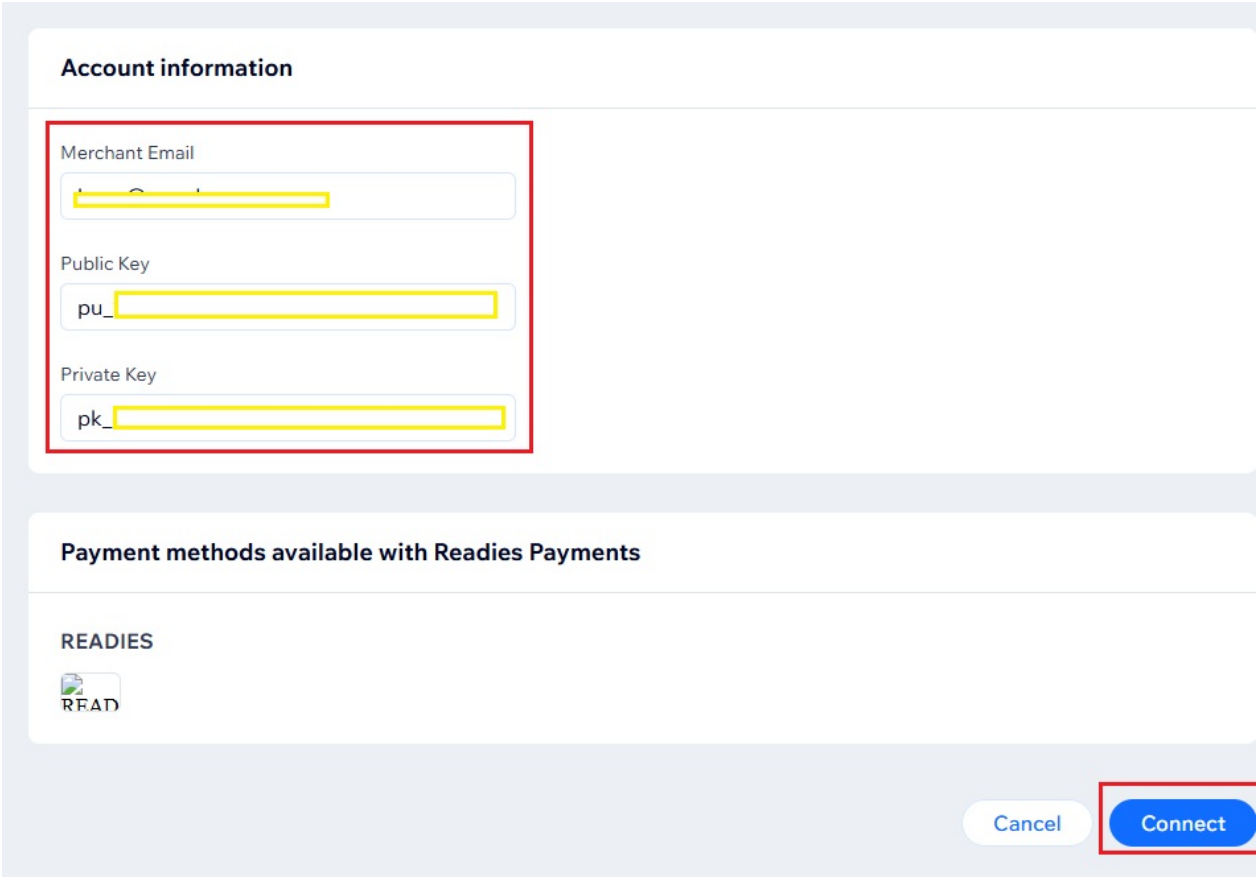
- Navigate to Accept Payments.



- Find {New Plugin} in the list of payment options.
- Click "Connect".



- Fill in all the required fields.



- Click "Submit" to complete the connection to Readies Payment Provider.